VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA
EKONOMICKÁ FAKULTA

KATEDRA APLIKOVANÉ INFORMATIKY

Implementace metodiky SCRUM v malé softwarové firmě

The implementation of SCRUM methodic in a small software company

Student: Aneta Jurczková

Vedoucí bakalářské práce: Ing. Jan Ministr, Ph.D.

Ostrava 2019

VŠB - Technická univerzita Ostrava
Ekonomická fakulta
Katedra aplikované informatiky

Zadání bakalářské práce

Student: **Aneta Jurczková**

Studijní program: B6209 Systémové inženýrství a informatika

Studijní obor: 6209R017 Informatika v ekonomice

Téma: Implementace metodiky SCRUM v malé softwarové firmě
The Implementation of SCRUM Methodic in a Small Software Company

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Úvod
2. Teoretická východiska agilního vývoje softwaru
3. Popis stávajícího stavu vývoje softwaru v dané softwarové firmě
4. Návrh implementace metodiky SCRUM
5. Závěr

Seznam použité literatury
Seznam zkratek
Prohlášení o využití výsledků bakalářské práce
Seznam příloh
Přílohy

Seznam doporučené odborné literatury:

OŠKRDAL, Václav a Petr DOUCEK. *Praktické řízení ICT projektů*. Praha: Oeconomia, 2014. 255 s. ISBN 978-80-245-2073-5.
STELLMAN, Andrew and Jennifer GREENE. *Learning Agile: Understanding Scrum, XP, Lean, and Kanban*. Sebastopol: O`Reilly Media, 2014. 397 p. ISBN 978-1-449-33192-4.
WIEGERS, Karl E. and Joy BEATY. *Software Requirements*. 3 ed. Washington: Microsoft Press, 2013. 672 p. ISBN 978-80-7356-7966-5.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Jan Ministr, Ph.D.**

Datum zadání: 23.11.2018
Datum odevzdání: 10.05.2019

Ing. Petr Rozehnal, Ph.D.
vedoucí katedry

prof. Dr. Ing. Zdeněk Zmeškal
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

V OSTRAVĚ dne 7.5.2019

Aneta Gruszková

jméno a příjmení

Poděkování

Ráda bych poděkovala panu Ing. Janu Ministrovi, Ph.D. za vedení této práce, za jeho čas, cenné rady a připomínky, bez kterých by tato práce nevznikla.

Obsah

1 Úvod

Vývoj informačních technologií se od poloviny minulého století neustále žene kupředu. Ve světě hardwaru a softwaru dochází k nekončícím inovativním změnám, na které musí organizace pružně reagovat a udržovat si svou konkurenceschopnost.

Agilní metodiky jsou v současnosti velmi populárními přístupy, jak vývoj softwaru přizpůsobit dnešní zrychlené a dynamické době. Firmy se potýkají s velkou konkurencí, a proto je zapotřebí držet krok, dodávat produkty kvalitně a v co nejkratším možném termínu. Aby si firmy mohly navzájem konkurovat, je nezbytné, aby svůj postup optimalizovali a modernizovali. K tomu jim poslouží právě zmiňované agilní metodiky. Pro tyto metodiky je charakteristická lepší organizace práce, rychlá a průběžná reakce na změnu požadavků a dokonalá spolupráce týmu. Jednou z nejrozšířenějších a nejčastěji používanějších agilních metodik je Scrum. Scrum je procesní rámec, který se používá k řízení vývoje od začátku devadesátých let. Název vznikl podle populární hry rugby, kde slovo scrum (zkratka „scrumage“, neboli „mlýn“) znamená shromáždění celého týmu za účelem získání nebo udržení míče. Autoři Scrumu jsou Ken Schwaber a Jeff Sutherland,

Cílem práce je navrhnout a implementovat agilní metodiku dle Scrum v softwarové firmě. Práce je rozdělená do tří částí, kde první část je teoretická, další dvě jsou praktické. Prvotní část obsahuje teoretická východiska dané problematiky. První kapitola obsahuje rozdíly mezi tradičním a agilním vývojem, ostatní metodiky, které využívají agilní vývoj a základní agilní principy. Následující kapitoly obsahují podrobný popis metodiky Scrum, jejich klíčových rolí, artefaktů a eventů. Druhá část sestává z charakteristiky a analýzy společnosti, na kterou je práce zaměřena. Část zahrnuje její předešlé zkušenosti se Scrumem, přiblížení problémů s poslední implementací, návrh řešení těchto překážek a na úplný konec návrh nejlepšího postupu pro implementaci této metodiky.

2 Teoretická východiska agilního vývoje softwaru

Počátkem devadesátých let, kdy se začaly počítačové technologie stávat nezbytnou součástí každého podniku, nastávaly s vývojem softwaru problémy. Krize tkvěla v pomalých reakcích na změny požadavků zákazníků, protože tradiční rigorózní metodiky přestávají v takových podmínkách vyhovovat. Projekty často překračovaly časový rámec, rozpočet nebo naopak funkcionalita nesplňovala zákazníkovy požadavky.

Tento zásadní problém se stal hlavním podnětem pro změnu. Začaly vznikat agilní metodiky vývoje softwaru. Jsou to metodiky, které jsou orientovány na zákazníka, a které vítají změny v průběhu vývoje. Agilní přístup klade velký důraz na spokojenost zákazníka a jeho začlenění do procesu vývoje. Zákazník je v tomto ohledu klíčovou součástí procesu. Zároveň se ale snaží eliminovat zbytečnou dokumentaci nevýznamných událostí či aktivit a zjednodušit procesy změn.

V únoru 2001 se sešli představitelé nových přístupů k vývoji softwaru, byla vytvořena Aliance pro agilní vývoj softwaru a podepsán Manifest agilního vývoje softwaru. V současné době si mohou projektoví manažeři vybrat z mnoha metod, které se zabývají životním cyklem vývoje softwaru tak, aby odpovídaly určitému projektu. Nejlepší volbou jsou agilní metodologie, které umožňují týmům reagovat na měnící se požadavky a nepředvídatelnost zákazníků prostřednictvím inkrementální i iterativní práce.

2.1 Porovnání rigorózních a agilních metodik

Rigorózní neboli tradiční metodiky a agilní metodiky jsou dvě skupiny metodik, které vycházejí z rozdílného chápání procesu vývoje softwaru.

Tradiční metodiky jsou velmi podrobné, formální a společně s referenčními modely procesů, různými modely životního cyklu, posuzováním zralosti a způsobilosti procesů jsou součástí tradičních přístupů budování informačních systémů. Přesně stanovují procesy, všechny požadavky a produkty. Lidský faktor je zde v roli zaměnitelné součástky. Tradiční metodiky předpokládají, že tvorbu IS lze přesně definovat, popsat a opakovaně realizovat. Většina těchto metodik je založena na vodopádovém modelu životního cyklu, ve kterém jednotlivé fáze vývoje softwaru následují po sobě. Velký důraz je také kladen na psaní důkladné psaní dokumentace.

Agilní přístupy jsou opakem tradičních. Vychází z předpokladu, že proces tvorby informačního systému nelze přesně popsat, má být pružný a má nabízet rychlá řešení. Nedefinují procesy, ale popisují jen principy a praktiky. Vychází ze zkušeností získaných během vývoje, kdy je třeba projekt přizpůsobovat aktuální situaci a reagovat na změny a na požadavky zákazníka. Schopnosti a dovednosti lidí jsou velmi důležitou kapitolou. Aby bylo možné agilní metodiky efektivně použít, musí být splněny určité předpoklady. Nejdůležitějším předpokladem použití agilních metodik je požadavek, aby zákazník byl součástí týmu a byl k dispozici podle potřeby.

2.2 Agilní metodiky vývoje softwaru

Agilní metodiky se dobře uplatňují na projektech, kde je třeba řešení vytvořit velmi rychle a pružně je přizpůsobovat měnícím se požadavkům zákazníka. Do agilních metodik byly od počátku zahrnovány tyto metodiky a přístupy:

Dynamic Systems Development Method (DSDM)

Metodika představuje rozšíření praktik rychlého vývoje aplikací. Za vznikem metodiky stojí celkem šestnáct organizací. Zajímavostí je, že autoři k této metodice dodávají i podpůrné vývojové prostředí (framework). Základní technikou, která se používá při analýze a návrhu, je prototypování. DSDM definuje tři hlavní fáze (Functional Model, Design and Build, Implementation), kterým předchází fáze Feasibility Study a Business Study. DSDM má velmi propracovaný systém školení a rozsáhlou dokumentaci.

Adaptive Software Development (ASD)

Metodika je pokládána jako filosofický základ agilních metodik a je považována za nejdynamičtější metodiku z rodiny, přesto však zachovává procesní přístup. Vychází z teorie komplexních adaptivních systémů. Jejím základem je změna a přizpůsobení změně.

Metodika postupuje iterativně, stejně jako u ostatních agilních přístupů. Na konci iteraci si ale neklade za cíl dodat fungující produkt, nýbrž pouze dodat zákazníkovi dostatek podkladů pro ověření postupu. Metodika ASD je léta prakticky používaná, a oproti většině agilních metodik byla úspěšně použita i u rozsáhlých projektů.

Feature-Driven Development (FDD)

Metodika Feature-Driven Development je agilní metodika, která definuje procesy a zdůrazňuje úlohu modelování při vývoji. Je založena na iterativním vývoji, který je řízen užitnými vlastnostmi produktu. FDD je považována za konzervativnější metodiku, protože má blíže ke klasickému přístupu.

Extreme Programming (XP)

Extrémní programování se zařazuje mezi jedny z nejznámější agilních metodik. Někdy jsou dokonce agilní metodiky a pojem XP mylně označovány jako ekvivalent.

Extrémní programování je metodika určená pro malé až středně velké týmy, které vyvíjejí software, jehož zadání se rychle mění, nebo není jasné. Metodika je velmi lehká, ale také disciplinovaná. Zavádí specifické praktiky jako párové programování, refaktorizace nebo například testy před kódováním.

Lean Development

Metodika je aplikací principů známých jako Lean Manufacturing a Total Quality Management na oblast vývoje softwaru. Cílem je vytvoření softwaru tolerantního ke změnám s třetinovou lidskou prací, s třetinovým časem, s třetinou investic do nástrojů a metod, s třetinovým úsilím přizpůsobit se novým podmínkám. Zatímco většina agilních metodik se zabývá taktickou úrovní, Lean Development se zaměřuje na strategickou úroveň ve vazbě na podnikovou strategii.

Scrum

Scrum je metodika zaměřená především na řízení projektu. Oproti XP je konzervativnější, tzn. na místo kolektivního vlastnictví je zodpovědnost definována za každý objekt či jeho množinu. Chápe procesy při vývoji softwaru jako empirické, které není možné předvídat, ale je nutné je monitorovat. Výhodou metodiky je především zásadní orientace na řízení týmu a důraz na řízení rizik v průběhu celého vývoje. Samotná metodika se rozšířila na začátku devadesátých let a u jejího vzniku stáli Ken Schwaber a Jeff Sutherland.

Scrum je metodika, které se budu věnovat v této práci, kde bude dopodrobna rozebrána. Na závěr bude uveden návrh implementace této poměrně známé a rozšířené metodiky.

Crystal metodiky

Crystal je rodina metodik, které jsou určeny pro různé typy projektů – pro projekty určité důležitosti a rozsahu. Všechny metodiky jsou postaveny na stejných hodnotách, kterými jsou komunikace a lehkost produktu. Jednotlivé prvky metodiky se upravují pro každý projekt. Tím se vyznačuje konfigurovatelnost této metodiky. Výběr metodiky z rodiny se provádí na základě velikosti projektu a důležitosti systému. Jednotlivé metodiky jsou pojmenovány podle barev, nejlehčí je Clear, potom následuje Yellow, Orange, Red, Maroon, Blue, Violet atd.

Agile Modeling

Agilní modelování je inspirováno metodikou Extrémní programování a aplikuje její praktiky, principy a hodnoty na oblast modelování. Agilní modelování není ucelená metodika, protože pokrývá jen oblast modelování. Může být použita jak v rámci rigorózních metodik, tak v rámci agilních metodik.

2.2.1 Základní principy agilního vývoje

Principy agilních metodik jsou formulovány v Manifestu pro agilní vývoj softwaru. V únoru 2001 byla vytvořena Aliance pro agilní vývoj softwaru a podepsán Manifest agilního vývoje softwaru. Manifest deklaruje čtyři hodnoty. Hodnoty na levé straně mají větší váhu než prvky na straně pravé. Manifest zní:

„Objevujeme lepší způsoby vývoje software tím, že jej tvoříme a pomáháme při jeho tvorbě ostatním. Při této práci jsme dospěli k těmto hodnotám:

Jednotlivci a interakce před procesy a nástroji.

Fungující software před vyčerpávající dokumentací.

Spolupráce se zákazníkem před vyjednáváním o smlouvě.

Reagování na změny před dodržováním plánu". (MANIFEST, 2001)

V tomto Manifestu je také popsáno 12 principů agilního vývoje. „Mezi tyto principy patří:

1. Naší nejvyšší prioritou je vyhovět zákazníkovi časným a průběžným dodáváním hodnotného softwaru.

2. Vítáme změny v požadavcích, a to i v pozdějších fázích vývoje. Agilní procesy podporují změny vedoucí ke zvýšení konkurenceschopnosti zákazníka.

3. Dodáváme fungující software v intervalech týdnů až měsíců, s preferencí kratší periody.

4. Lidé z byznysu a vývoje musí spolupracovat denně po celou dobu projektu.

5. Budujeme projekty kolem motivovaných jednotlivců. Vytváříme jim prostředí, podporujeme jejich potřeby a důvěřujeme, že odvedou dobrou práci.

6. Nejúčinnějším a nejefektnějším způsobem sdělování informací vývojovému týmu z vnějšku i uvnitř něj je osobní konverzace.

7. Hlavním měřítkem pokroku je fungující software.

8. Agilní procesy podporují udržitelný rozvoj. Sponzoři, vývojáři i uživatelé by měli být schopni udržet stálé tempo trvale.

9. Agilitu zvyšuje neustálá pozornost věnovaná technické výjimečnosti a dobrému designu.

10. Jednoduchost – umění maximalizovat množství nevykonané práce – je klíčová.

11. Nejlepší architektury, požadavky a návrhy vzejdou ze samo-organizujících se týmů.

12. Tým se pravidelně zamýšlí nad tím, jak se stát efektivnějším, a následně koriguje a přizpůsobuje své chování a zvyklosti.“ (MANIFEST, 2001)

2.3 Metodika SCRUM

Původně byl Scrum vyvinut pro řízení a vývoj produktů. Od počátku devadesátých let se intenzivně využívá zejména pro výzkum a identifikaci trhů, technologií a produktových možností, pro vývoj produktů a jejich vylepšení, vydávání produktů, vývoj a údržbu v Cloudu a dalších operativních prostředí pro produktové použití, údržbu a obnovu produktů. Scrum je založen na teorii empirického řízení procesů neboli empirismu. Dle empirismu, vycházejí znalosti ze zkušeností a rozhodovat bychom se měli na základě toho, co je známo. Každou implementaci podpírají tři pilíře: transparentnost, prozkoumávání a přizpůsobení.

Transparentnost

Významné aspekty procesu musí být viditelné a zřejmé pro osoby, které jsou odpovědné za výsledek. Aspekty musí být definovány sdílenými zásadami, díky nimž pozorovatelé chápou to, co vidí. Všichni účastníci proto musí sdílet společný jazyk týkající se procesu a sdílet společnou definici toho, co je považováno za „hotovo” („done”).

Přezkoumávání (INSPECT)

Uživatelé Scrumu jsou nuceni často přezkoumávat scrumové artefakty a postup směrem ke sprintovému cíli za účelem včasného odhalení nežádoucích stavů. Přezkoumávání by však

nemělo být tak časté, aby se stávalo překážkou v práci. Proto je nejproduktivnější, když přezkoumávání provádějí k tomu kvalifikovaní lidé, nejlépe v místě práce.

Adaptace (ADAPT)

Pokud se při přezkoumávání zjistí, že se alespoň jeden bod procesu odchyluje mimo přijatelné meze, nebo že výsledný produkt nebude přijatelný, musí se provést adaptace procesu. Aby se zmenšila pravděpodobnost vzniku jakékoliv další odchylky je třeba úpravu provést v co možná nejkratší době.

Vývoj probíhá v 30denních iteracích nazývaných Sprint, ve kterých se dodává vybraná množina užitných vlastností. Klíčovou praktikou jsou denní porady (Scrum Meetings), které slouží pro koordinaci a integraci prací.

Metodika Scrum je rozdělena do 4 fází:

Plánovací fáze (planning phase)

Specifikace požadavků, plánu dodávek, architektonické a obchodní vize.

Vynášecí fáze (staging phase)

K požadavkům se připojují požadavky, které nemají vliv na funkčnost.

Fáze vývoje (development phase)

Tým dodává funkcionalitu s nejvyšší prioritou, na konci každé iterace (sprintu) předvede výsledek.

Fáze dodávky (release phase)

V poslední fáze se předá hotový produkt uživatelům.

Během jednoho sprintu se členové týmu zapíší k úkolům a pracují společně na splnění cíle. Každý den se účastní denních porad (Scrum Meeting). Tyto porady monitorují stav projektu a včas identifikují problémy a překážky, které mohou ohrozit úspěch projektu. Konají se každý den, ve stejnou dobu a optimálně trvají 15 minut. Porady řídí Scrum Master a účastní se jich všichni členové týmu, dokonce i manažeři. Každý člen týmu musí zodpovědět tři otázky: které úkoly na projektu dokončil od minulé porady, které nové úkoly na projektu má řešit a jaká může očekávat omezení při řešení úkolu.

2.3.1 Projektový tým

Scrum metodika specifikuje pouze tři oficiální role, které hrají hlavní úlohu v týmu: Product Owner, vývojový tým a Scrum Master. Tito lidé sdílejí různé úkoly a odpovědnosti související s dodáním produktu. Scrum týmy můžeme charakterizovat jako self-organizing a cross-functional, tzn. že se týmy samy organizují a jsou schopny pokrýt různé funkční oblasti.

Scrum Master

Scrum metodika definuje podstatnou roli s názvem Scrum Master, která je v týmu skutečně vyžadována, ostatně jako všechny tři hlavní role Scrumu. Scrum Master není klasický teamleader nebo manažer, ale spíše prostředník mezi týmem a jakýmikoliv rušivými elementy zvenku. Je odpovědný za propagaci a podporu Scrum metodiky. Scrum Master to dělá tím, že pomáhá všem zúčastněným pochopit teorii, praxi, pravidla a hodnoty metodiky.

Scrum Master především pomáhá týmu dosáhnout jeho cíle, odstraňuje problémy, motivuje k lepším výsledkům a chrání tým před nežádoucími vlivy zvenčí. Měl by být komunikativní a vnímavý. Je součástí týmu a je mu také kdykoli k dispozici, proto by měl sedět s týmem v jedné místnosti. Role Scrum Mastera by se nikdy neměla kombinovat s jinými definovanými rolemi, tzn. že například člen vývojového týmu by neměl zastávat funkci Scrum Mastera už jen z toho důvodu, že by upřednostňoval svoji práci před ostatními. Stejně tak roli Scrum Mastera nesmí zastávat Product Owner. Nejlepší variantou je, aby se této roli věnoval někdo na plný úvazek.

Pokud Scrum Master vykonává svoji práci dobře, nedá se na jeho pozici nahlížet jako na náklad navíc. Tím, že zajistí vyšší efektivitu, kvalitu a motivaci, ušetří v celkovém rozsahu více peněz, než jaký náklad na Scrum Mastera musí podnik vynaložit.

Product Owner

Další z rolí, které Scrum zavádí, je role Product Ownera. Product Owner můžeme jednoduše označit jako vlastníka produktu. Tato osoba má na starost definování vize projektu a její transparentní komunikaci mezi vývojovým týmem a zákazníkem. Product Owner definuje priority, rozhoduje, na které funkcionalitě se bude pracovat dříve, na které později a na které třeba vůbec. Má na starosti Business Value a také ROI celého produktu. Tato role je pro úspěch celého procesu klíčová, proto vyžaduje komunikativního člověka s hodnotnými znalostmi produktu.

Product Owner je také zodpovědný za tvorbu Product Backlogu a zároveň je týmu pravidelně podle potřeby k dispozici. Hlavním cílem je porozumění produktu a porozumění požadavkům a hodnotám, které zákazník očekává. Jak už bylo zmiňováno, Product Owner by měl být týmu i zákazníkovi k dispozici, často se však dostává do situace, kdy je od týmu příliš daleko. Zpravidla se tak stává ve velkých korporacích, které mají týmy po celém světě. Tým potom často nemá k Product Ownerovi přístup. Aby se těmto situacím předcházelo, obvykle ze svého středu najde někoho, kdo má k produktu a prostředí zákazníka nejblíže, ten pak zastává roli Product Owner Proxy. Product Owner Proxy musí být schopen nejen Product Ownera zastoupit a odpovídat na otázky týmu, ale i činit rozhodnutí a nést za tato rozhodnutí zodpovědnost.

Vývojový tým

„Vývojový tým se skládá z profesionálů, kteří na konci každého sprintu doručují potenciálně vydatelný přírůstek „hotového“ („Done“) produktu.“ (Schwaber a Sutherland, 2017, s.7)

Vývojový tým je strukturovaný a oprávněný organizovat a řídit si práci sám (samoorganizující). Skládá se z členů s různými funkcemi, kteří jsou schopni dosáhnout cílů sprintu. Velikost týmu je značně omezená, velikost totiž musí zachovat flexibilitu a zároveň zvládnout dokončit veškerou práci v rámci sprintu.

Role zákazníka v projektu

I když se do projektového týmu nezahrnuje, je velmi důležitým účastníkem projektu (mimo Product Ownera). Zákazník je pro projekt velmi podstatný, protože je to právě on, pro koho se celý projekt vytváří, a proto můžeme říct, že zákazník je součástí týmu dle potřeby. „Scrum je založen na tom, že se zákazníkem vývojářský tým úzce spolupracuje.“ (Myslín, 2016, s.81)

Metodika Scrum definuje tři pilíře (transparentnost, přezkoumávání, adaptace), které napomáhají k zapojení zákazníka do projektu a přizpůsobivost ke změnám požadavků.

Existují dva druhy zákazníka: ten, kdo software objednává a platí (právní zákazník) a ten, kdo software bude používat a kdo z něj bude těžit (faktický zákazník).

2.3.2 SCRUM Eventy

Agilní přístupy pouze žádá, abychom se učili ze svých chyb a zajisté identifikovali nové způsoby, jak se zlepšit. Jako jeden z principů Manifest uvádí:

"Tým se pravidelně zamýšlí nad tím, jak se stát efektivnějším, a následně koriguje a přizpůsobuje své chování a zvyklosti."

Právě díky této výzvě k otevřené komunikaci nás Scrum povzbuzuje, abychom v průběhu sprint pořádali pět klíčových událostí, které nám pomohou efektivně a úzce spolupracovat, stejně jako zlepšit naše znalosti i do budoucna. Všechny události jsou zásadně důležité, a to je důvod, proč krátce popíšu každou z nich.

Sprint Planning

Plánování sprintu je událost, která vždy spouští nový sprint a kde Product Owner a vývojový tým diskutují o tom, které položky produktového Backlogu (Product Backlog Items, PBI) zahrnou do sprintu. Zatímco Product Owner má právo upřednostňovat každou položku Backlogu do sprintu, vývojový tým je připraven reagovat a v případě potřeby některé položky přesunout na jiný sprint. Vývojový tým potom předpovídá, kolik položek může v zadaném sprintu dodat, vzhledem ke schopnostem a k faktorům týmu, které by mohly čas i zdroje, které mají k dispozici, ovlivnit.

Účelem takového meetingu, kde dojde k naplánování sprintu, je získat Sprint Goal a Sprint Backlog, které všichni odsouhlasí, a které je realistické a dosažitelné.

Daily Scrum

Scrum usiluje o efektivní využití času i zdrojů a událost Daily Scrum není výjimkou. Daily Scrum je časově omezený (z angl. time-boxed) na pouhých 15 minut. Může se to zdát málo, ale mnoho vývojových týmů považuje tuto techniku za velmi užitečnou. Denní Scrum je příležitost, která se nabízí vývojovému týmu, aby zhodnotil svůj pokrok při dosahování cíle sprintu, přehodnotil a naplánoval své aktivity pro tento den.

Sprint

„Jádrem Scrumu je sprint v délce trvání jednoho měsíce či méně, během nějž je vytvořen „done", použitelný a potenciálně vydatelný přírůstek produktu." (Schwaber a Sutherland, 2017, s.10)

Jako každá Scrum událost, má i sprint časově stanovenou délku, která je většinou v rámci celého vývoje shodná. Po skočení předchozího sprintu začíná vždy Sprint nový. Sprint je limitovaný hlavně proto, aby nedocházelo ke vzrůstu rizik jako je zvýšení složitosti nebo přírůstku k cílovému produktu, které mohou nastat z důvodu příliš dlouhého trvání jednoho takového sprintu.

Sprinty se skládají z plánování sprintu (Sprint Planning), denních meetingů (Daily Scrums), práce vývojového týmu, vyhodnocení sprintu (Sprint Review) a sprintové retrospektivy (Sprint Retrospective).

Sprint Review

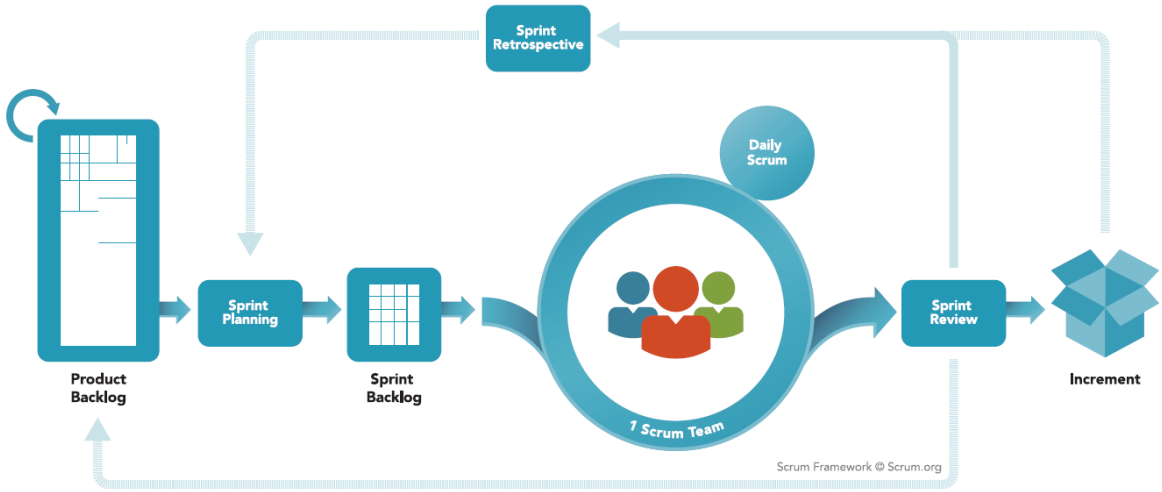
Na konci sprintu se koná Sprint Review. Jedná se o čtyřhodinové setkání s časovým rozvrhem, kdy tým představí Product Ownerovi a všem ostatním zúčastněným stranám (Stakeholders) to, co bylo během sprintu vyvinuto.

Sprint Retrospective

Poslední událostí je Sprint Retrospective, která nastane vždy na konci sprintu, obvykle trvá asi hodinu. Účastníky jsou vývojový tým, Scrum Master a Product Owner. Slovo „agilní" je o neustálém zlepšování a rychlém získávání zpětné vazby tak, aby se samotný produkt a jeho vývoj stále zlepšovaly. Retrospektiva pomáhá týmu pochopit, co funguje dobře a co ne. Neustálé zlepšování je to, co udrží a řídí vývoj a k tomu jsou retrospektivy klíčové. Sprintová retrospektiva je prostředkem k určení fungujících částí, tím pádem může vývojový tým diskutovat, spolupracovat a hledat kreativní řešení vyskytlých problémů.

2.3.3 SCRUM Artefakty

Metodika Scrum definuje artefakty, které zajišťují maximální transparentnost informací. Reprezentují práci nebo hodnotu. Jsou speciálně navrženy tak, aby všichni chápali artefakty stejně.



Obr. 2.1 - The Scrum Framework

Produktový backlog

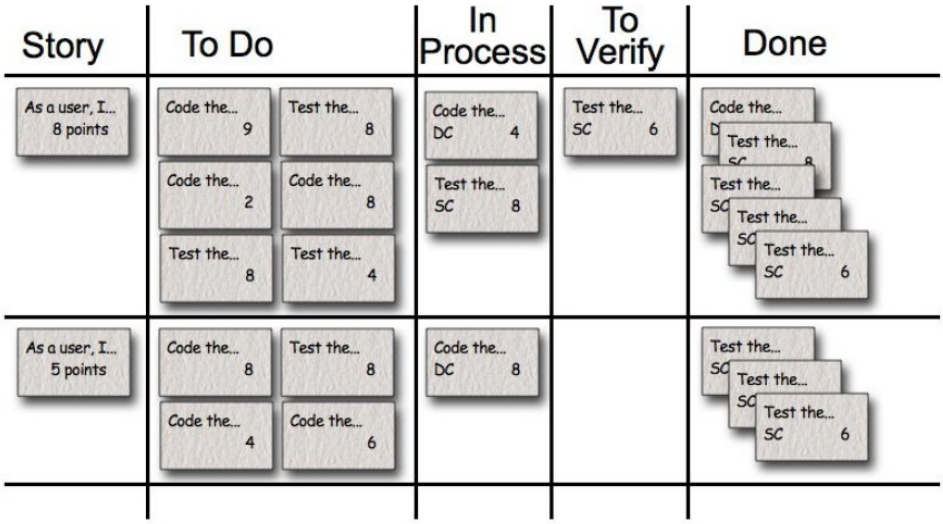
Produktový Backlog (Product Backlog) je soupis všech vlastností, funkcí, požadavků a rozšíření, které jsou pro vývoj produktu klíčové. Do produktového backlogu se přidávají funkcionality, které by měl výsledný produkt mít. Každá položka, která je uvedená v backlogu, odpovídá jedné funkcionalitě, kterou má produkt obsahovat. Položky sepsané v produktovém backlogu mají popis, prioritu a odhad časové náročnosti. Produktový backlog je takzvaně „živý“ artefakt, protože se jeho položky neustále mění. Na začátku projektu obsahuje produktový backlog jenom ty nejdůležitější a nejvíce požadované funkcionality. V průběhu vývoje se backlog rozšiřuje o další funkcionality.

Product Owner spolupracuje s vývojovým týmem a společně tvoří produktový backlog obsahující detailně popsané položky, ke kterým se snaží odhadnout náročnost. Každá položka musí být zrealizovatelná nejpozději do konce jednoho sprintu. „Product Owner je zodpovědný za obsah, dostupnost a prioritizaci backlogu. Vývojový tým je odpovědný za všechny odhady.“ (Schwaber a Sutherland, 2017, s.15-16)

Sprintový backlog

Sprintový backlog (Sprint backlog) je soubor položek z produktového backlogu, které byly vybrány do sprintu, včetně plánu na dodání a splnění cíle. Sprintový backlog má prezentovat určitou funkčnost, která má být dodána na konci iterace jako přírůstek produktu. Vývojový tým předpovídá, kolik práce bude k dodání této funkčnosti přírůstku potřeba. Sprintový backlog náleží vývojovému týmu a jen ten má právo rozvržení práce měnit.

Vývojový tým backlog během sprintu upravuje podle vlastní potřeby a tím ho neustále utváří. Děje se tak hlavně proto, že tým během vývoje získává poznatky o tom, co je potřeba pro splnění aktuálního sprintu. Pokud je potřeba zařadit nový úkol do backlogu, tak jej tým zařadí a upraví odhad zbývající práce. Podobný postup se děje při zjištění úkolu, který je zbytečný. Takovýto úkol se odstraní a tým opět odhaduje zbývající čas. „Sprintový backlog tak představuje vysoce jasný, transparentní a aktuální obraz práce, kterou vývojový tým plánuje v průběhu sprintu dokončit.“ (Schwaber a Sutherland, 2017, s.17)



Obr. 2.2 - Sprint Backlog

User Story

User story je uživatelský jednoduchý a krátký popis toho, co by měl výsledný produkt dělat, bez toho, jak by to měl dělat, z pohledu osoby, která si přeje novou funkcionalitu. Obvykle se jedná o uživatele nebo zákazníka. User Story je hodnoceno podle své náročnosti určitými Story Points.

Tvar User Story: Jako <uživatel>, chci <funkcionalitu>, abych <dostal hodnotu>.

User Story má tři základní části:

Definice role

User Stories a role jsou spolu svázány. User Story má smysl pouze tehdy, pokud pro sebe má i roli. Nejprve proto definujeme roli, až poté User Story, které se to týká.

Definice cíle

Zde je nutno popsat, co chceme se systémem vykonávat za činnost. Popis musí být výstižný, věcně a přímo definovat specifikaci úkolu.

Definice užitku

Je to to, čeho chceme dosáhnout. Ve skutečnosti nepovinná část User Story.

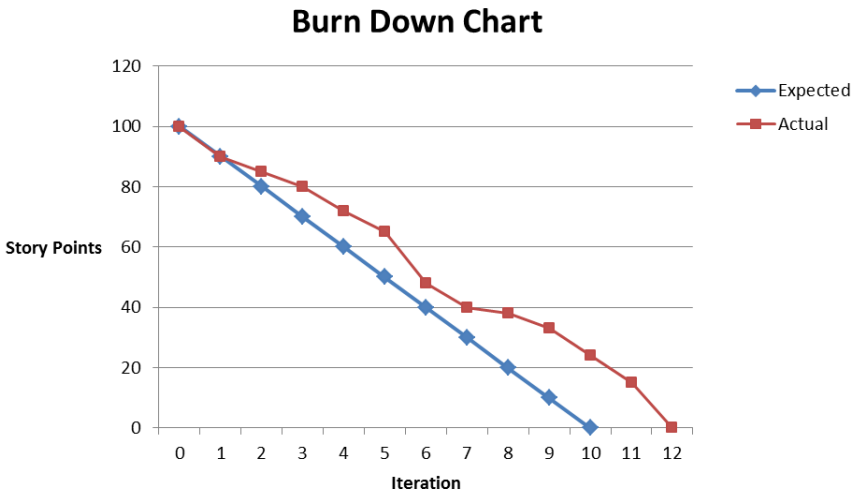
Jednou z výhod agilních User Stories je, že mohou být napsány v různých úrovních detailů. Příliš rozsáhlé popisy, které pokrývají velké množství funkcí, označujeme jako eposy „epics“. Protože jsou eposy obecně velké na to, aby je vývojový tým zvládl dokončit v jedné iteraci, je rozdělen právě na několik menších User Stories předtím, než je zpracován. User Stories mají pomoci týmu a Product Ownerovi v soustředění na zákazníka.

Přírůstek

Přírůstek neboli Increment je součet všech položek produktového backlogu dokončených během aktuálního sprintu v kombinaci s přírůstky všech předchozích sprintů. Přírůstek na konci sprintu musí být ve funkčním stavu bez ohledu na to, zda se Product Owner rozhodne tento přírůstek vydat.

Burndown diagram

Scrum využívá speciální druh grafu, který je označován jako Burndown diagram. Jedná se o spojnicový graf, který ukazuje množství zbývající práce v rámci jednoho sprintu. Zbývající práci můžeme sledovat v jednotkách story point. Osa Y zobrazuje množství práce, kterou zbývá dokončit do konce sprintu. Osa X znázorňuje čas, intervalem je jeden sprint. První hodnotou na ose Y je množství práce pro celý sprint. Při vývoji produktu se vytváří křivka, která určuje zbývající čas.



Obr. 2.3 - Burndown Chart

Ve kterémkoliv okamžiku můžeme vidět, ve které fázi sprintu se nacházíme a kolik času do konce sprintu zbývá. Vývojový tým, Product Owner nebo zákazník může vidět pokrok v plnění úkolů a rychlost tohoto pokroku.

3 Popis stávajícího stavu vývoje softwaru v dané softwarové firmě

Vzhledem k přání majitele, obsahuje tato část pouze základní informace, aby nedošlo k porušení anonymity společnosti.

V této kapitole si na začátek firmu charakterizujeme, analyzujeme současný stav spolu se snahou o nasazení metodiky, shrneme si nedostatky a problémy při vlastní implementaci a požadavky na změny, které firma předkládá k novému návrhu. Následně navrhneme řešení nalezených problémů a požadavky na změny, které vedení společnosti předložilo. Tyto poznatky využijeme k návrhu implementace metodiky Scrum v následující kapitole.

Zprvu je ale nutné upozornit na fakt, že metodika Scrum je pouze procesní rámec, tudíž ovlivňuje pouze vybrané aspekty procesu vývoje softwaru. Scrum se neprojevuje v některých projektových disciplínách jako například ve finančním plánování, v řízení rizik nebo přidělování členů k projektu. Scrum je tedy procesní rámec, do kterého lze zahrnout procesy, techniky, nástroje a metodiky. Mnohdy je metodika Scrum spojována s další agilní metodikou XP.

3.1 Charakteristika společnosti

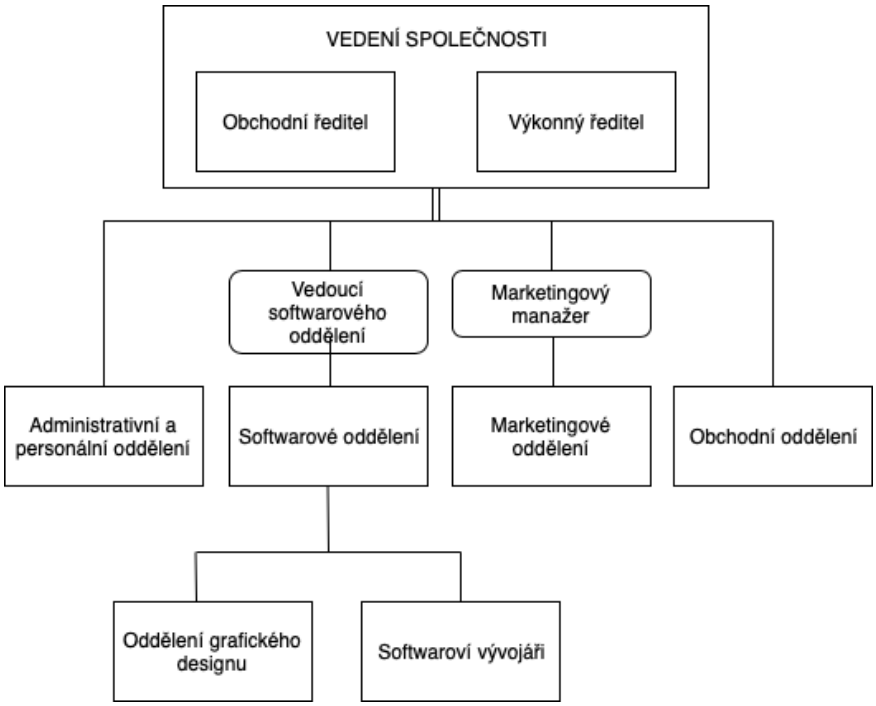
Jedná se o softwarovou společnost, která byla založena roku 1994, s prvotním plánem prodeje a servisu výpočetní techniky. Podle definice pro určení velikosti podniku dle počtu zaměstnanců je firma považována za malý podnik, který zaměstnává méně než 50 zaměstnanců. V současné době se firma soustředí na více odvětví v oblasti IT, hlavně pak na vývoj zakázkového a vlastního softwaru.

Firma začínala jako malá prodejna spojená se službami prodeje a servisu hardwaru, následně i softwaru. Společnost se začala postupně rozrůstat o nové zaměstnance a nabídku produktů a služeb. Okruh činností se rozšiřoval a společnost začala směřovat jiným směrem, od prodeje a servisu k vývoji a prodeji vlastního, na míru vytvořeného produktu. V dnešní době se převážně specializuje na zakázkový vývoj softwaru a vývoj vlastních softwarových produktů.

Firma je ve všech směrech dynamická, moderní a inovativní, v minulosti byla mnohokrát oceněna a vlastní několik certifikací. Také je hrdým partnerem několika větších českých i nadnárodních společností. Vedení usiluje o nestálý růst v oblasti informačních technologií. Snaží se své softwarové produkty vyvíjet vždy kvalitně s přihlédnutím na veškeré požadavky zákazníka, aby byly splněny i ty nejmenší požadavky.

3.1.1 Organizační struktura společnosti

Firma je vedena dvěma společníky. Jeden z nich zastává funkci výkonného ředitele, druhý obchodního ředitele. Výkonný ředitel je zodpovědný za řízení organizace a směr, kterým se vyvíjí. Má na starosti strategii a plánování cílů, které se firma snaží plnit. V otázce dalšího směřování má hlavní slovo i přes to, že se nachází ve stejné hierarchické rovině jako obchodní ředitel. Obchodní ředitel se soustředí na obchodní strategii. Ovlivňuje inovační a vývojové procesy, dohlíží nad strategickým řízením s ohledem na interní zdroje a obchodní politiku společnosti. Oba společníci spravují čtyři různá oddělení. Každé oddělení má svého vedoucího, který odpovídá za správný chod svého resortu.



Obr. 3.1 - Organizační struktura společnosti (vlastní zpracování)

3.2 Analýza současného stavu

Scrum již v dnešní době není cizí slovo. Patří mezi nejpoužívanější metodiky agilního řízení projektu a mnoho. Firma se již dříve usilovala o nasazení metodiky Scrum do svého vývoje, avšak bez úspěchu. Scrum není principálně složitý, můžeme o metodice říct, že je relativně jednoduchá. Jednoduchá metodika ale neznamená, že je jednoduché ji správně použít.

Společnost současně realizuje projekty, které rozlišujeme podle velikosti. Velikost projektu určuje množství lidské práce, čas a finance. Projekty ve firmě rozdělujeme na projekty menšího, malého, většího a víceletého rozsahu. Na menších projektech pracuje 1 programátor, na kterého dohlíží projektový manažer. Tento druh projektu většinou trvá jeden den až třicet dní, jedná se o aktualizaci nebo menší změnu v již existujících projektech.

Malé projekty jsou řešeny jedním až třemi programátory s dohledem projektového manažera. Tyto projekty trvají maximálně čtyři měsíce. Projekty většího rozsahu trvají obvykle jeden rok. Na projektech pracuje jeden projektový manažer a až tři programátoři. Firma realizuje i víceleté projekty, kdy na projektech pracuje tři až šest programátorů s dohledem projektového manažera.

Způsob, jakým firma současně realizuje tyto projekty, bychom mohli přirovnat k tradičnímu vodopádovému modelu. V dnešních podmínkách, které konkurence a zákazník klade, jde zcela o nevyhovující model, proto je nutné implementovat agilní metodiku Scrum správně, užitečně a samozřejmě včas.

3.3 Dřívější snaha o nasazení metodiky Scrum

První snaha společnosti o agilní vývoj nedopadla úspěšně. Nebyla ani tak devastující, byla jen neužitečná. Scrum není principiálně složitý, metodika je relativně jednoduchá, jenomže jednoduchá metodika neznamená, že je jednoduché ji správně použít. I přes to, že byly zavedeny všechny techniky, které Scrum definuje, vývoj softwaru nedopadl tak, jak bylo původně očekáváno. Vedení společnosti bylo přesvědčeno, že se agilní způsob vývoje nehodí do jejich prostředí a agilní metodiku opustilo. Později však zaznamenali pár chyb, které mohly ovlivňovat nově zavedený agilní přístup. Hlavní důvody, proč firma přestala metodiku uplatňovat, jsou shrnuty níže.

3.3.1 Identifikované problémy během první implementace

1. **Nedodržování základní scrumové techniky Daily Scrum.**

   Hlavní problém nastal tehdy, kdy členové týmu přestali vnímat Daily Scrum jako užitečný. Některé členy týmu taková schůze zdržovala. Začali meeting brát jako čas, který jim někdo krade nad naplánovanou prací. Přestali sdělovat jistá očekávající omezení nebo nedostatky, aby urychlili strávený čas na schůzi. Později se již nikdo o schůzi moc nezajímal. Schůze přestaly být pravidelné, začaly se rušit. Takže když někdo z týmu neměl na Daily Scrum čas, prostě a jednoduše tuto schůzi zrušil. To mělo za následek nevědomé zdržování týmu, natahování termínů a později bohužel i nesplnění přání zákazníka.

2. **Problémy v komunikace mezi členy týmu**

   Členové mezi sebou přestali komunikovat, začali se soustředit čistě jen na svou práci a svá zadání, aby předešli vysvětlování a objasňování druhým. Opět začalo docházet k nedodržování termínů a sprity se protahovaly. To vše jen díky nedostatku komunikace a spolupráce týmu. Zvyšovalo se riziko vzniku chyb a problémů. Přestalo se myslet kolektivně a týmově. Jednotliví členové se začali zajímat jen o své povinnosti, a to nemělo dobré dopady na tým.

3. **Nedostatečná spolupráce a snaha Product Ownera**

   Product Owner přestal hájit zájmy zákazníka, který o produkt žádal a zároveň nezodpovědně přiděloval práci týmu, který se na vývoji produktu podílel. Software přestával plnit požadavky, které plynuly ze strany zákazníka. Zákazník přestával být spokojený a tým prodlužoval strávený čas na projektu. Pod vyvíjeným tlakem se začaly objevovat chyby v kódu, které se špatně opravovaly. Vývojový tým ztratil přehled o částech kódu a celý projekt se řítil do propasti.

2. **Nedostatek zkušeností**

   Tým nemá bohužel žádné zkušenosti s žádným podobným typem vývoje. Někteří špatně nebo neúplně pochopili danou problematiku. Neměli možnost se dostatečně připravit a metodiku pochopit. Náběh na agilní vývoj byl příliš rychlý. Nebyly zodpovězeny dotazy, které zúčastnění měli. Členové přestali metodiku vnímat jako užitečnou a prospěšnou.

3.4 Požadavky na změnu

Cílem práce je úspěšná implementace metodiky Scrum do prostředí firmy. Dosavadní způsob vývoje je neužitečný a již nesplňuje požadavky firmy. Firma by chtěla předejít problémům z minulosti a tentokrát implementovat správně a účinně.

Hlavní požadavky na změny vedení společnosti sepsalo do přehledné tabulky 3.1.

Tab. 3.1 - Seznam požadavků (zpracováno vedením společnosti)

Pořadí	Požadavek	Očekávání
1.	Omezit riziko vyšších nákladů	Předvídatelnost a kontrola nad projektem. Pozorovat pokroky ve vývoji.
2.	Rychlé odezvy na změny zákazníka	Zákazníci by měli mít možnost měnit svá zadání a požadavky na výsledný produkt. Posílit spokojenost zákazníka.
3.	Zapojení zákazníka do vývoje	Diskutovat se zákazníkem na jeho potřebách, požadavcích během vývoje tak, aby se předešlo nesplnění cíle.
4.	Celkové zapojení týmu	Aby se každý člen týmu začal zapojovat do realizace projektu. Nikoli jen vedení.

4 Návrh implementace metodiky SCRUM

Na základě analýzy předchozí kapitoly bude vytvořen návrh implementace metodiky Scrum, tedy pouze jako procesního rámce. První podkapitola obsahuje návrh na řešení problémů, se kterými se firma v počátcích setkala, spolu s doporučením praktik a agilních principů potřebných ke správnému zavedení Scrumu do firmy. Návrh implementace obsahuje kroky ke správné implementaci a bude uveden v druhé podkapitole. Je rozdělen do etap, které jsou vyobrazeny v diagramu. Navržení procesního rámce bude popsáno stručně, podrobnější popis je uveden v teoretické část práce.

V předešlé kapitole jsme se zaměřili na současný stav vývoje softwaru ve společnosti a odkryli jsme základní problémy, se kterými se firma při první implementaci potýkala. U každého již definovaného problému se pokusíme vytvořit návrh na jeho řešení. Definujeme, ve kterých částech se může firma zlepšit a kde se podle analýzy nachází největší nedostatky.

4.1 Řešení problémů

1. **Nedodržování základní scrumové techniky Daily Scrum**

   Daily Scrum je každodenní, 15minutová schůzka, určená k synchronizaci aktivit týmu a vytvoření plánu na dalších 24 hodin. Může se zdát, že se jedná o zcela zbytečnou ztrátu času, ale to není úplně pravda. Daily Scrum je jedním z nejvíce podceňovaných a obvykle špatně prováděných Scrum meetingů. Představuje totiž každodenní kontrolu a koordinaci týmu, plánování společné práce a identifikaci překážek, které během plnění cíle mohou nastat. Při krátkých schůzkách si tým připomíná, na čem jeho členové pracují a s čím se potýkají. Z takto probraných činností může tým lehce definovat, kde může nastat chyba, která by mohla ohrozit vývoj produktu. Správně prováděný Daily Scrum vede ke zvyšování efektivity a produktivity, proto je důležité tuto techniku dodržovat.

   Je velmi důležité, aby Scrum Master řádně vysvětlil podstatu meetingů a objasnil Scrum týmu, proč je provádění nutné pro dosažení cíle. Pokud se dodrží veškerá pravidla a zavede se povinnost docházky, náplň meetingu by měla být přínosná pro všechny členy týmu. Tým musí pracovat společně, organizovaně. Členové musí dbát na zájmy týmu, a nejen na zájmy své. Musí začít brát Daily Scrum jako meeting, který je ve vývoji posune dál, dál jako tým, dál jako jedince.

2. **Ztráta komunikativnosti**

Jedním z klíčových faktorů fungujícího systému je komunikace. Pokud se ve vývoji objeví překážky a členové Scrum týmu přestanou komunikovat, může takovýto postup vést k větším problémům, které se budou jen stěží opravovat. Komunikace je základním pilířem metodiky, protože staví právě na komunikaci mezi zákazníkem a vývojovým týmem. Aby byly splněny všechny požadavky zákazníka, musí tým a Product Owner pracovat jako jeden, být sehraní a navzájem si důvěřovat. Jak už bylo řečeno v předešlých kapitolách, metodika Scrum umožňuje během vývoje agilně reagovat na změny, se kterými se ale naopak počítá. Během vývoje s použitím agilní metodiky dochází k neustálé výměně informací, které vedou ke splnění cíle a požadavků. Komunikace musí probíhat mezi jednotlivci, týmy, ale i organizací jako celkem. Firma také nesmí zapomínat na eventy, které Scrum definuje, protože ty vedou k dokonalé komunikaci mezi všemi zúčastněnými.

3. **Nedostatečná spolupráce a snaha Product Ownera**

Product Owner je součástí vývojového týmu. Dohromady mají jeden cíl, a to dodat hodnotu zákazníkovi. Měli by pracovat společně jako jeden. Společně přemýšlet a rozhodovat.

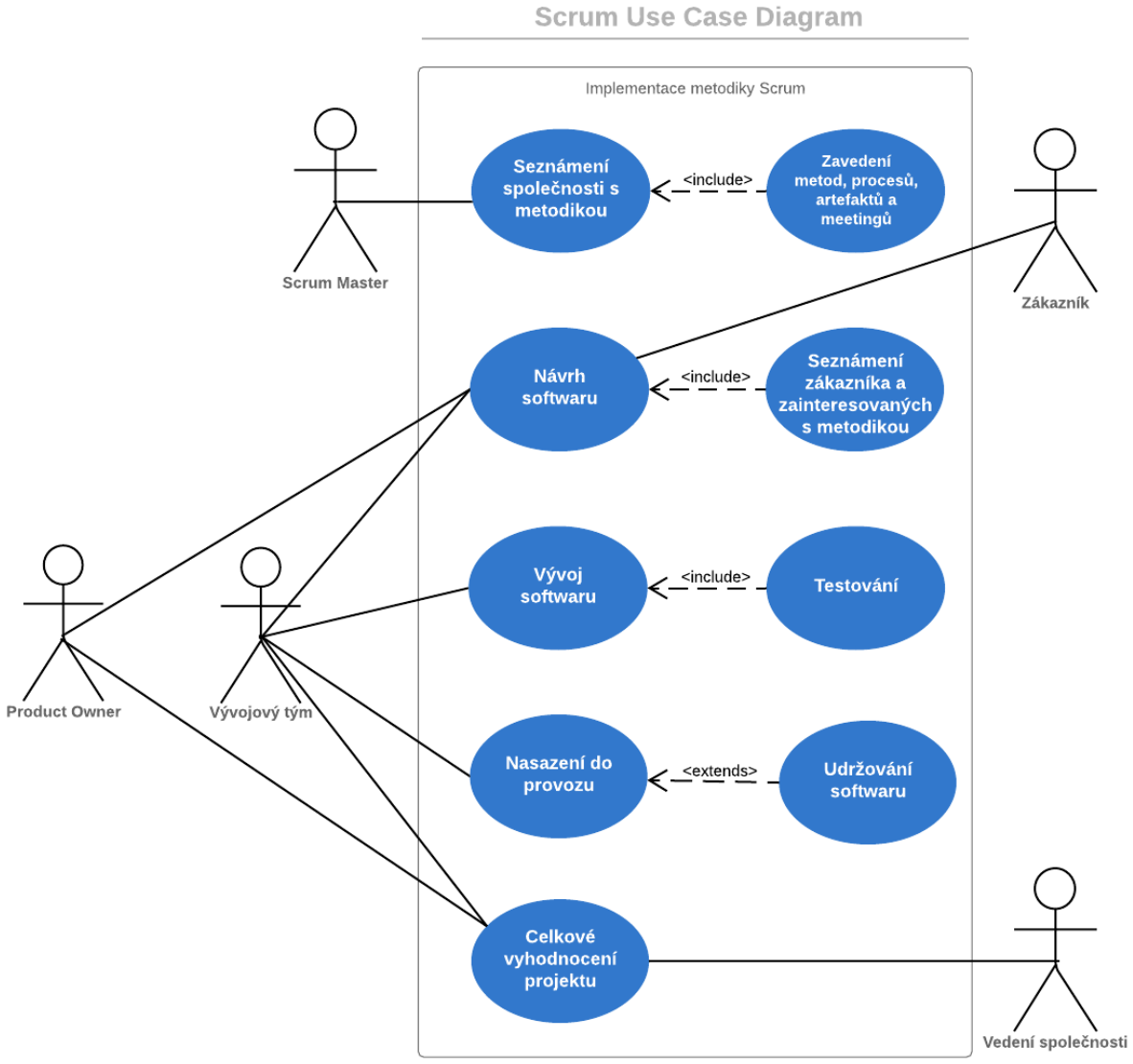
Product Owner je zodpovědný za jasné vyjadřování, řazení a prioritizaci položek v produktovém backlogu. Jeho práce také zahrnuje zajištění, že vývojový tým chápe dané položky. Product Owner se stává klíčovým článkem firmy, protože jeho správné fungování přináší maximalizaci hodnoty produktu. Musí tým vést a směřovat ho k dosahování naplánovaných cílů tak, aby byli spokojeni jak zákazníci, tak i samotná organizace. Aby byla práce Product Ownera vykonávaná úspěšně, musí jeho rozhodnutí respektovat celá firma.

4. **Nedostatek zkušeností**

Nedostatečná zkušenost jako problém vzniká bohužel při každém novém startu. Nezáleží na tom, zda se jedná o první implementaci metodiky do firmy nebo o výrobu zcela nového produktu. Nedostatek zkušeností postihuje všechny začínající, v této firmě tomu není jinak. Plno začínajících firem očekává ohromující výsledky hned po zavedení nových technik, jenomže bez řádného pochopení agilních principů, nebude jejich teorie fungovat. Je podstatné, aby byla agilní metodika pochopena. Také spolupráce se zkušeným člověkem v této oblasti, může firmě pomoci předejít komplikacím, které by jinak bezesporu nastaly.

4.2 Návrh implementace

Návrh implementace agilní metodiky je rozčleněn do pěti hlavních etap. Návrh je zahájen seznámení s metodikou spolu se zavedením veškerých potřebných náležitostí, následným návrhem a vývojem softwaru, nasazením do provozu a konečným celkovým vyhodnocením projektu. Tyto etapy a jejich uživatelské role jsou vyobrazeny v Use Case diagramu 4.1.



Obr. 4.1 - Scrum Use Case diagram (vlastní zpracování)

4.2.1 Seznámení s metodikou

Při tvorbě návrhu implementace je ze všeho nejdůležitější seznámit všechny zúčastněné s metodikou a ujistit se, že všichni dané problematice rozumí. Jeden z hlavních problémů, který

je popsán v předešlé kapitole, byla nedostatečná připravenost týmu na metodiku. Pro firmu je významné vědět, že všichni, kteří se budou na projektu podílet, rozumí této metodice, všem jejím praktikám a principům, které zavádí. Zmíněnou povinnost má na starosti také Scrum Master. Společnost se taky musí rozhodnout, zda bude seznámení s metodikou Scrum provádět pomocí externího zprostředkovatele, nebo pověří osobu přímo z firmy. Výhody jsou na obou stranách. V případě zvolení proškolení externí firmou, je velkou výhodou jejich zkušenost a praxe. Na druhou stranu může toto školení zvednout neplánované náklady. Pravým opakem je internista, tedy osoba z firmy, která sice žádné zkušenosti nemá, což může být velmi riskantní, za to je ale méně nákladná. Seznámení s metodikou Scrum bude probíhat formou meetingu, které se musí předem naplánovat. Pokud je tým řádně proškolen, můžeme začít přidělovat role a definovat jejich pravomoc, povinnosti a míry zodpovědnosti. Role, které musí firma zavést:

- **Scrum Master**

  Osoba, která všem pomáhá pochopit scrumovou teorii, praxi, pravidla a hodnoty. Vede tým k dosahování cílů, odstraňuje překážky, koučuje a motivuje tým. Chrání ho před vnějšími vlivy, které by mohly bránit v jeho postupu. Měl by podporovat tým i jednotlivce v rozvoji. Být komunikativní, vnímavý, ale i utlumovat případné konflikty, které v týmu nastanou. Měl by být týmu kdykoliv k dispozici, a proto by měl sedět s týmem v jedné místnosti. Tuto roli by měl zastávat specialista, který této metodice dokonale rozumí a splňuje její vlastnosti.

- **Product Owner**

  Česky jej můžeme nazvat jako vlastník produktu. Zajišťuje, aby cíle, rozsah a doména produktu byly všemi členy pochopeny. Efektivně spravuje produktový backlog tak, aby maximalizoval získaný přínos. Na rozdíl od Scrum Mastera už nesedí v jedné místnosti s týmem. Svůj čas tráví se zákazníky, aby dostatečně porozuměl jejich prostředí a dokázal vždy rozeznat pravou hodnotu pro zákazníka. Product Owner neřídí tým ani jednotlivé neřídí, stanovuje jen to, co se má vytvořit a v jakém pořadí – určuje priority. Roli Product Ownera by neměla zastávat osoba s rolí Scrum Mastera. Roli bude zastávat jedna konkrétní osoba z firmy v závislosti na projektu.

- **Vývojový tým**

  Jedná se o tým, kteří na konci každého sprintu prezentují přírůstek produktu. Vývojový tým je oprávněn sám sebe organizovat a řídit si vlastní práci. Tým se skládá z osmi programátorů, z toho jeden architekt. První čtyři programátoři budou zaměřeni na back end, zbývající tři na frond end. Vývojový tým bude své postřehy konzultovat se Scrum Masterem, který by jim měl být kdykoliv k dispozici.

- **Zákazník**

  Nepostradatelná součást projektu, bez které by vývoj produktu neměl smysl. Podle představ zákazníka je vyvíjen produkt, který je následně u zákazníka implementován. Není součástí projektového týmu, k tomu slouží jako mezičlánek Product Owner.

4.2.2 Zavedení metod, procesů, artefaktů a meetingů

Po sestavení projektového týmu můžeme začít zavádět jednotlivé metody, procesy a artefakty. Ve firmě musíme zavést tyto artefakty:

- **Produktový Backlog**

  Product Owner vytvoří backlog spolu se zákazníkem. Jsou zde popsány veškeré funkcionality, které je nutné pro zákazníka vytvořit. Funkcionality budou ve formě User Stories. Product Owner a zákazník přiřadí všem funkcionalitám prioritu.

- **Sprintový Backlog**

  Sprint Backlog má podobu Produktového Backlogu, nicméně v něm nejsou zahrnuty všechny funkcionality, nýbrž jen ty, které musí vývojový tým za daný Sprint vytvořit. Obsažené funkcionality se vybírají vždy na začátku Sprintu. Výběr si tvoří Scrum tým.

- **User Stories**

  Do User Stories jsou zapsány funkcionality, které musí vystihovat pravý účel k dosažení přidané hodnoty produktu.

- **Burndown graf**

  Graf slouží pro názorné zobrazení průběhu projektu. Tým pomocí softwarového nástroje uvidí, kolik času na projektu zbývá a co už bylo splněno.

Po zavedení artefaktů je nutné zavést meetingy, které tato metodika také specifikuje. Mezi meetingy, které se do firmy zavádí, patří:

- **Daily Scrum**

  Každodenní patnácti minutová schůzka, která probíhá vždy na stejném místě. Měla by sloužit k synchronizaci aktivit a tvorby plánu na další den. Odhaduje se množství práce do další schůzky a zároveň se kontroluje odvedená práce z té minulé. Každý člen týmu odpovídá na tři základní otázky. Tým může takto vyhodnotit postup práce, který směřuje k cíli Sprintu. Organizátorem schůzky je Scrum Master, který dohlíží na průběh. Schůze by se měla konat vždy ráno v předem stanovený čas, který by měl zůstat neměnný. Díky těmto schůzkám by firma měla předcházet nepříjemným překážkám během celé průběhu vývoje.

- **Sprint Review**

  Sprint Review je vyhodnocení, které se provádí vždy na konci daného sprintu a zabývá se výsledným přírůstkem produktu. Během vyhodnocení tým a zúčastněné osoby probírají výsledky Sprintu. Všichni zúčastnění diskutují o následném pokračování směrem k dosažení hodnoty. Jedná se o neformální schůzku, u Sprintů, které trvají měsíc, by měla trvat nanejvýš čtyři hodiny. V případě kratších Sprintů, je i tato schůze kratší. Pokud to situace vyžaduje, může toto vyhodnocení vést k přizpůsobení produktového backlogu.

- **Sprint Planning**

  Product Owner a vývojový tým mají za úkol diskutovat o tom, které položky produktového Backlogu zahrnou do aktuálního sprintu. Product Owner má právo upřednostňovat každou položku backlogu. Vývojový tým na tuto prioritizaci Product Ownera reaguje a v případě potřeby některé položky přesouvá na jiný sprint. Musí také předpovídat, kolik položek může v zadaném sprintu dodat tak, aby daný sprint mohl dokončit v rámci svých schopností a s ohledem na jeho zkušenosti a možnosti. Na konci diskuze dojde

k naplánování sprintu. Hlavním cílem této schůzky je získat Sprint Goal a Sprint Backlog, které následně všichni odsouhlasí.

- **Sprint Retrospective**

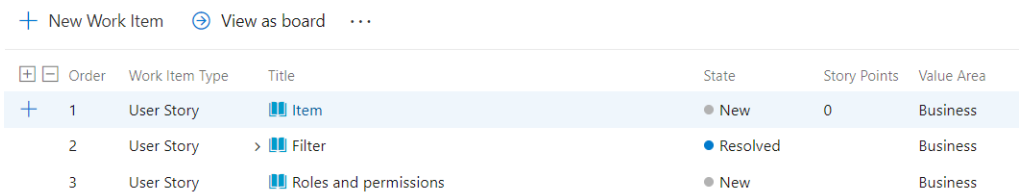
  Tento meeting poslouží k prozkoumání průběhu posledního sprintu. Tým identifikuje slabé stránky a uvažuje o potenciálním vylepšení. Retrospektiva napomáhá Scrum týmu naplánovat kroky, které povedou ke zlepšení v následujícím Sprintu. Scrum Master schůzku pořádá a zajišťuje, aby všichni zúčastnění chápali její účel. Na schůzce je přítomen jako člen týmu se zodpovědností za scrumový proces. Povzbuzuje tým, aby usiloval o zlepšení svých vývojových procesů a postupů tak, aby byl příští Sprint efektivnější.

4.2.3 Seznámení zákazníka a zainteresovaných s metodikou

Nyní je zapotřebí seznámit zákazníky se specifiky nové metodiky a s tím, jak bude práce na projektu probíhat. Product Owner probere se zákazníkem základní cíle, požadavky a specifika projektu. Zákazník musí upřesnit, jak by měl produkt vypadat a co by měl splňovat. Poté co je Product Owner seznámen se zákazníkovými požadavky, začne tvořit produktový backlog. Plánuje se první meeting, kde se vytvoří projektový plánu s údaji o rozpočtu, nákladech a délce projektu.

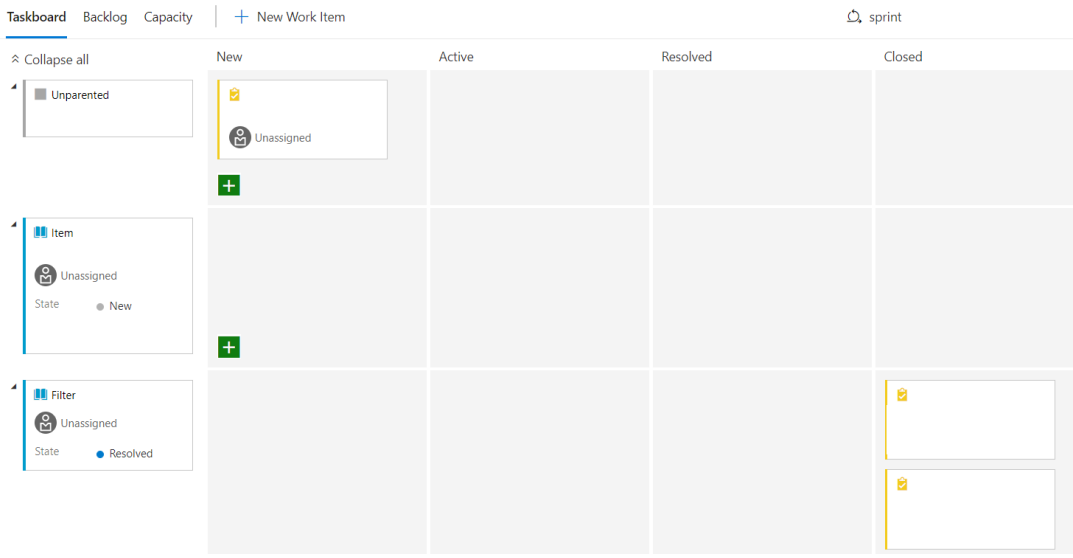
4.2.4 Návrh a vývoj softwaru

Po naplánování probíhá první sprint se všemi náležitostmi, které byly uvedeny. Během prvního sprintu většinou dochází k vytvoření grafického návrhu, architektury systému a datového modelu. První sprint je ze všech ostatních sprintů nejkratší, u menších projektů může trvat jen pár dní.



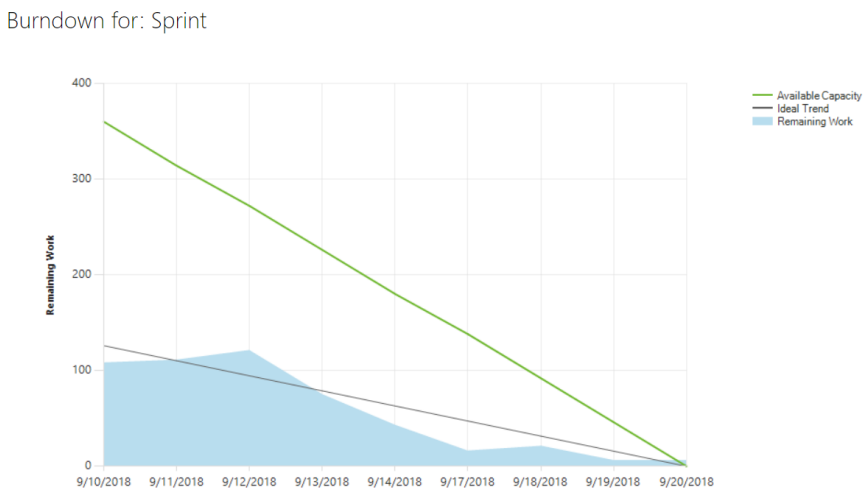
Obr. 4.2 – User Stories projektu (vlastní zdroj)

Vzniká první meziprodukt, který slouží zákazníkovi k nahlédnutí a kontrole. Zákazník kontroluje, jakým směrem se produkt vyvíjí a do jaké míry splňuje jeho požadavky. Po kontrole může vývoj odsouhlasit a začíná další nový sprint. Může taky návrh zamítnout, v takovém případě firma musí návrh přehodnotit a vytvořit znovu.



Obr. 4.3 - TaskBoard konkrétního Sprintu (vlastní zdroj)

Následuje fáze implementace, kde se realizuje samotný projekt. V průběhu následujících sprintů dochází k přírůstku na produktu, jeho testování a po splnění všech požadavků a přání zákazníka i následné nasazení. Postup v rámci Sprintů sledujeme v Burndown grafu.



Obr. 4.4 - Burndown graf pro konkrétní Sprint (vlastní zdroj)

Softwarový nástroj

Společnost si pro vývoj zvolila software již dříve, při svých pokusech o první implementaci. Jednalo se o rozšíření Microsoft Visual Studia s názvem Visual Studio Team Services, který se na konci minulého roku přejmenoval a změnil na novou sadu Azure DevOps služeb. Microsoft Azure je neustále se rozvíjející sada cloudových služeb, která napomáhá firmě usnadňovat práci. DevOps zahrnuje jednoduché a spolehlivé nástroje pro průběžné doručování. Umožňuje vývojovému týmu přidělovat speciální role nebo využívat nástroje pro softwarové architekty, developery a testery.

Výhodou této sady služeb je variabilita výběru, kterou sada Azure DevOps nabízí. Firma si může zvolit, které služby jsou pro práci přínosné a které naopak ke svému vývoji nepotřebuje. Na výběr jsou služby:

- *Azure Boards* – možnost plánovat, sledovat a diskutovat o práci v rámci týmu,
- *Azure Repos* – nabízí neomezené privátní úložiště Git v cloudu,
- *Azure Test Plans* – testování s využitím sady nástrojů,
- *Azure Pipelines* – možnost průběžně sestavovat, testovat a nasazovat projekt na libovolnou platformu a cloud,
- *Azure Artifacts* – vytváření a sdílení balíčku s týmem,
- *Azure Monitor* – služba představuje úplný přehled o aplikacích, infrastruktuře a síti.

Společnost se rozhodla využívat všechny uvedené služby na základě dřívější analýzy potřeb. Tato cloudová služba obsahuje veškeré agilní nástroje s podporou metodiky Scrum, které jsou pro vývoji softwaru tímto způsobem charakteristické.

4.2.5 Nasazení do provozu a celkové vyhodnocení projektu

V rámci nasazení musí proběhnout test kompatibility s jinými systémy. Po nasazení by měla společnost a vedoucí vývojového týmu zajistit proškolení zaměstnanců zákazníka, aby se software stal přínosným pro všechny. Na úplný závěr projektu proběhne celkové vyhodnocení průběhu, splnění cíle, termínů a profitu z projektu.

4.2.6 Udržování softwaru

Tímto však starost společnosti, která vyvíjela software, nekončí. Je zapotřebí, aby se nasazený software udržoval, popřípadě jeho modernizoval nebo aktualizoval na přání zákazníka. Změna je vždy vítaná, proto by se tým následných úprav obávat neměl.

5 Závěr

Společnost, na kterou je práce zaměřena, se zabývá vývojem webových aplikací a realizací webových stránek. Specializuje se především na zakázkový vývoj softwaru a vývoj vlastních softwarových produktů. Společnost se už dlouho rozhoduje pro změnu ve vývoji produktů. Po prvním nezdařeném pokusu implementace se zdálo, že firma už nadále o agilního přístup ve vývoji zájem mít nebude. Avšak tradiční styl řízení projektů nevyhovuje moderním podmínkám. Firma je nucena efektivně a agilně řídit projekty, aktivně reagovat na požadavky zákazníka, proto je nezbytné dosavadní přístup změnit.

Primárním cílem práce bylo navrhnout implementaci agilního rámce Scrum do současného zažitého vývoje v menší softwarové firmě. Zprvu bylo nutné věnovat pozornost teorii, abychom si přiblížili danou problematiku. Teorií se zabývala první část práce. Byly zmíněny pojmy, které Scrum definuje. Přiblížili jsme si rozdíly mezi agilním a tradičním přístupem ve vývoji, stručně jsme si popsali ostatní agilní metodiky a poté jsme se významněji zaměřili na konkrétní metodiku Scrum. Rámec Scrum je v dnešní době velmi oblíbenou a vyhledávanou variantou, jak se přizpůsobit současné rychlé a dynamické situaci ve světě. Objasňuje efektivnost řízení produktů a postupů k vylepšování produktu, týmu, ale i prostředí. Definovali jsme role, ceremonie a artefakty, které pravidla Scrumu popisují.

Ze získaných poznatků byla vypracována praktická část. Praktická část začíná charakteristikou firmy. Abychom lépe pochopili fungování společnosti a dokázali navrhnout implementaci Scrumu, museli jsme firmu analyzovat. V analýze jsme nalezli čtyři problémy, které dříve neumožnily hladký průběh implementace tohoto rámce. Tyto problémy jsme si podrobně popsali a vytvořili jsme návrh řešení těchto problému, který sloužil k následnému navržení implementace jako příklad, čeho se vyvarovat. Veškeré informace vycházely z konzultací s vedením firmy.

Metodika Scrum má nesmírnou výhodu v kladné reakci na změny v zadání a požadavcích, které jsou nesmírnou výhodou pro zákazníka, ale taky velkým konkurenčním krokem pro aplikující firmu.

6 Seznam použité literatury

[1] OŠKRDAL, Václav a Petr DOUCEK. Praktické řízení ICT projektů. Praha: Oeconomia, 2015. 255 s. ISBN 978-80-245-2073-5.

[2] STELLMAN, Andrew and Jennifer GREENE. Learning Agile: Understanding Scrum, XP, Lean, and Kanban. Sebastopol: O`Reilly Media, 2014. 397 p. ISBN 978-1-449-33192-4.

[3] WIEGERS, Karl E. and Joy BEATY. Software requirements. 3ed. Washington: Microsoft Press, 2013. 672 p. ISBN 978-80-7356-7966-5.

[4] MYSLÍN, J. Scrum: průvodce agilním vývojem softwaru. Brno: Computer Press, 2016. 167 s. ISBN 978-80-251-4650-7

[5] BUCHALCEVOVÁ, A. Metodiky budování informačních systémů. 1.vyd. Praha: Oeconomica, 2009. 205 s. ISBN 978-80-245-1540-3

[6] BUCHALCEVOVÁ, Alena. Metodiky vývoje a údržby informačních systémů. 1. vyd. Praha: Grada, 2005. 163 s. Management v informační společnosti. ISBN 80-247-1075-7.

[7] ŠOCHOVÁ, Z. a E. KUNCE. Agilní metody řízení projektů. Brno: Computer Press, 2014. 176 s. ISBN 978-80-251-4194-6.

[8] THE AGILE ALLIANCE. Manifest Agilního vývoje software. Agilemanifesto.org [online]. 2001 [cit. 8.1.2019]. Dostupné z: http://www.agilemanifesto.org/iso/cs/

[9] SCRUM ALLIANCE. The Scrum Guide. ScrumGuides.org [online]. 2017 [cit. 15.1.2019]. Dostupné z: https://www.scrumguides.org/scrum-guide.html

[10] KNIBERG, Henrik. Scrum and XP from the Trenches: How We Do Scrum. C4Media Inc., 2007. 140 s. ISBN 978-1-4303-2264-1.

[11] COHN, Mike. Agile estimating and planning. Upper Saddle River, NJ: Prentice Hall Professional Technical Reference, 2006. 368 s. ISBN 0131479415.

[12] KADLEC, Václav. Agilní programování: metodiky efektivního vývoje softwaru. Vyd. 1. Brno: Computer Press, 2004. 278 s. ISBN 8025103420.

[13] KIM H. PRIES, Kim H.Jon M. Scrum project management. Boca Raton, FL: CRC Press, 2011. ISBN 9781439825174.

[14] DANIŠKO, Kristián. Implementace agilního projektového řízení ve start-up firmě. Ostrava, 2016. Diplomová práce. VŠB-Technická univerzita Ostrava, Ekonomická fakulta.

7 Seznam zkratek

Tab. – Tabulka

Obr. - Obrázek

Prohlášení o využití výsledků bakalářské práce

Prohlašuji, že

- jsem byla seznámena s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo;
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevýdělečně, ke své vnitřní potřebě, bakalářskou práci užít (§ 35 odst. 3);
- souhlasím s tím, že bakalářská práce bude v elektronické podobě archivována v Ústřední knihovně VŠB-TUO. Souhlasím s tím, že bibliografické údaje o bakalářské práci budou zveřejněny v informačním systému VŠB-TUO;
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona;
- bylo sjednáno, že užít své dílo, bakalářskou práci, nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

V Ostravě dne 7. 5. 2019

.....

jméno a příjmení studenta

Seznam příloh